11. Risk

Program Name: Risk.java   Input File: risk.dat

You are playing a classical game of RISK with another one of your friends. Since these games are long and boring, and you just want to know the outcome and be done, you pray to God that He may fix the seed of the universe's random number generator to the one you give him, so that you can predict how the game will play out. You program a simple computer simulation that plays each confrontation out as follows:

Both players start with a certain number of troops. One player is the attacker and one is the defender. Multiple "rounds" are played until one player is out of troops. Each round consists of multiple dice rolls. The attacker may roll a = min(3, n) die, where n is the number of troops he has, and the defender may roll b = min(2, n) die. Then, the top min(a, b) rolls from each player are placed in pairs, with the best from each player going in the first pair, and the second best from each player going in the second pair (if both rolled 2 dice). For each pair, if the attacker's roll is greater than the defender's roll, the defender loses a troop. But if the defender's roll is greater than or equal to the attacker's roll, the attacker loses a troop. For example, if the attacker has 2 troops and the defender has 1, the attacker rolls 2 dice, the defender rolls 1. Then, the attacker's best die and the defender's only die are compared, and a troop is removed using the above algorithm.

Your goal is, for a given random number seed, and the number of troops for each player at the beginning of the battle, to output the number of troops left for each player at the end of the battle. For the purposes of simulation, assume the attacker rolls all of his dice first, then the defender rolls all of theirs for every round.

Input
The first line will be a single integer N, the number of battles to simulate.
Each battle will consist of two lines. The first line will be a single long integer that is the seed for the random number generator you are to use. The second line is two integers A and D, or the number of troops for the attacker and the defender at the start of the battle. To make use of the Random object, initialize it with the seed given by saying `Random r = new Random(seed);`
A die roll is uniformly distributed over the range [1, 6]. Remember, however, that the Random object only generates a number in the range [0, N] with `r.nextInt(N + 1).`

Output
For each battle, output the number of troops left for the attacker, followed by the number of troops left for the defender.

Example Input File
```
3
1234
8 8
4484
3 5
10
1 1
```
Example Output to Screen
```
0 6
2 0
0 1
```